Portable Language-Independent Adaptive Translation from OCR

Final Report (Phase 1)

Contractor: **BBN Technologies**
10 Moulton Street
Cambridge, MA 02138

Principal Investigator: **Mr. Prem Natarajan**
Tel: 617-873-5472
Fax: 617-873-2473
Email: prem@bbn.com

Reporting Period: **21 November 2007 – 30 April 2009**

This material is based upon work supported by the
DARPA/IPTO
Portable Language-Independent Adaptive Translation from OCR
MADCAT Program
ARPA Order No: X 103
Program Code: 7M30
Issues by DARPA/CMO under Contract #HR001-08-C-0004

1. Introduction

The objective of MADCAT is to produce a robust, highly accurate transcription engine that ingests documents of multiple types and produces English transcriptions of their content. For addressing the technical challenges implicit in that goal, the BBN-led team proposed a system that embodies integration of five major operations: (1) pre-processing and image enhancement, (2) page segmentation, (3) text recognition, and (4) metadata extraction. In Phase 1 of the MADCAT effort, we made significant improvements in all the above areas. In addition, we developed an end-to-end system for processing the Phase 1 evaluation data. The evaluation system exceeded the Phase 1 program goal of 40% accuracy on 70% of the documents. Below, we summarize the work performed by the BBN-led team in Phase 1 of the MADCAT effort. We highlight our accomplishments by each technical area and also indicate the performers in that area.

2. Accomplishments in Pre-processing [BBN, Polar Rain, SUNY, UMD]

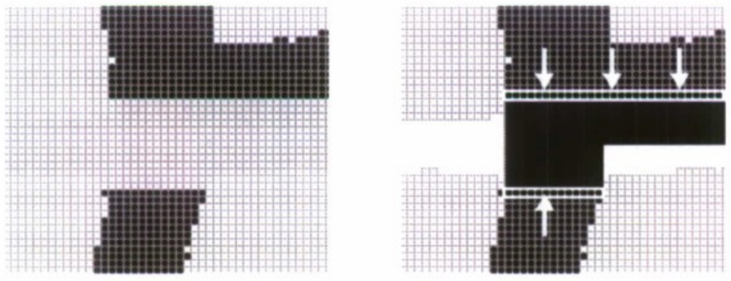
The goal of pre-processing is to enhance noisy/degraded images for text recognition and translation. Typically, handwritten text often contains non-text artifacts such as ruled page-lines, stains, speckle noise, etc. Often the document has breaks in the text glyphs and some glyphs are merged. Also the scanning may produce images at a low resolution.

In this phase we focused on two areas: (a) ruled page-line removal, and (b) image-enhancement.

2.1 Page Line Detection [BBN, Polar Rain, SUNY, UMD]

Heuristics for Page-line Detection and Removal [BBN]: In this approach, the input image is initially divided into ten equal vertical strips. Page-lines are detected for each strip as follows:

1. The projection profile of the intensity is normalized by dividing it by the width of the strip. The normalized projection profile is denoted by PROJ(*Y*).
2. A smoothing template is applied to the projection profile. The smoothed value PROJ_SMOOTHED(*Y*) at a given *Y* is the average of all the values within a window of *W* pixels wide, centered at *Y*. Different values of *W* (set to 2**dpi*/300 and 4**dpi*/300) are used to detect lines of different widths.
3. We search the smoothed projection profile for pixels *Y** such that PROJ_SMOOTHED(*Y**) > 0.5 and the local maxima is within the radius of 12*dpi*/300
4. For each *Y** detected, we search within $|Y - Y^*| < 3 \cdot dpi / 300$ of the strip for black pixels and mark them as "page-line pixels"



(a) Ruled-line detection (b) Classification of ruled-line pixels

Figure 1. Classification of ruled-line pixels as black or white.

The detected ruled-lines are classified as "black" or "white" heuristically. If a pixel not associated with the page-line is black and adjacent to a page-line pixel, then the pixels starting from the one immediately adjacent to the page-line pixel to the pixel at the center of the page-line are classified as "black". The rest of the pixels are classified as "white". An illustration is shown in Figure 1.

MRF-based Page-line Removal [BBN]: Most approaches for page-line removal such as the one described above introduce artifacts such as breaks in a glyph, which does not result in any significant improvement in word error rate (WER). To reduce the artifacts introduced by page-line removal, we developed a page-line removal and restoration algorithm using Markov Random Field (MRF). A binarized image is modeled as the output of an MRF and the pixels associated with the ruled-line are restored using the belief propagation algorithm. As shown in samples from MADCAT data (distributed by LDC) in Figure 2, our approach removes the page-line while still preserving the smooth edges in the handwritten glyphs. The MRF approach is also visually better than a heuristic-based approach for restoration.

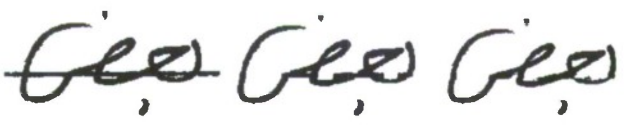

(a) Input (b) Heuristic (c) MRF

Figure 2: MRF based ruled-line removal.

Shape-DNA based Page-line Removal [Polar Rain]: In this phase, we designed a text line detection and removal algorithm for handwritten document images based on Shape-DNA models. In this approach, the binary input image is projected onto a text database. A ruled page-line database is computed from page lines, and the document image is projected onto this database in order to detect page lines. Shape-DNA models are thus computed off-line using page-line images (e.g., images full of horizontally parallel lines) so as to build a database that models the characteristics of page lines that are observed in handwritten document images. We compute a list of shape-DNA patterns that have similar shape to ruled line artifacts and exclude these patterns during projection onto the text database. We also compute the text database from only handwritten text images.

Input document images are then projected onto Shape-DNA models. The projection image is a probability image that shows line segments on the locations of possible page/rule lines and also on the locations of text characters. On the locations of possible page/rule lines, probabilities are much stronger compared to text characters. Therefore after binarizing the probability image, the images with and without page/ruled lines are obtained for each document. High level-semantic based morphological operators are applied to the projection distance image. The line detection process takes about two seconds for an image of size 2500x3300 pixels. Experimental results with more than a thousand images showed that the false positive rate (i.e., percentage of documents without rule lines that are detected to have lines) is less than one percent and the false negative rate is zero, In other words, all handwritten documents with rule lines are detected.

We also developed a page-line cleaning algorithm for handwritten documents with ruled lines. While projecting the input image onto Shape-DNA models, image blocks where the projection distance to the line database is smaller than a pre-defined threshold (e.g. twice the block length), are set to white patterns assuming that they correspond to rule lines, not text characters. To remove existing artifacts in this image, such as page-lines between characters, we exploit the line image and use the computed locations of page-lines to clear them from the page-line removed image. After page-line cleaning, the image may still have some line artifacts, such as small line segments that are close or connected to text characters. We have developed a handwritten image restoration technique in order to remove such artifacts and restore the shape of text characters. We do this by computing new Shape-DNA models using handwritten document images to model the text characters for handwritten text.

Run-length based Page-line removal [SUNY]: Many Arabic characters have long strokes that are almost horizontal and run along ruled page-lines. So, simple projection profile-based methods for ruled-line removal do not work well for such documents. In this phase, SUNY developed a module for page-line removal based on the following principles. Horizontal run-length smearing is used to fill in small gaps in ruled lines that were typically caused due to binarization artifacts. Next, an adaptive fuzzy run-length based approach coupled with heuristics is used to detect the position of the page-lines and subsequently remove the page lines. When ruled page-lines are removed, some pixels belonging to handwritten text strokes could also be inadvertently removed. Therefore, we used a region-growing based method to restore the text strokes damaged during page-line removal.

Page-line Detection and Removal using Linear Sub-spaces [UMD]: We investigated a novel method for removing page rule lines in monochromatic handwritten Arabic documents using subspace methods with minimal effect on the quality of the foreground text. We use moment and histogram properties to extract features that represent the characteristics of the underlying rule lines. During the training phase, we incrementally construct linear subspaces representing horizontal and vertical lines using a set of rule line-only images. During the testing phase, we measured the distance between features extracted from the test image and the previously constructed subspace. Pixels that belong to foreground text regions have larger distances to the subspace and will be left unchanged. We also introduced a pixel-level evaluation methodology that can generally be used to assess the accuracy of any noise removal algorithm and we applied it to the page rule line removal problem. Evaluating algorithms for noise removal from handwritten documents requires pixel-level annotation (i.e. ground truth) in order to compute meaningful statistics. Such pixel level annotation cannot be manually obtained for large collections of document images. In order to address this problem, we introduced a method for creating semi-synthetic data sets that can be used to evaluate noise removal algorithms. A number of pages that contain page-lines but no text are first scanned and binarized to create a set of page templates. These templates are then combined with a set of document images containing only text handwritten by different writers, in a cross product fashion. Interactions between text (in different positions and styles) and ruled lines results in significant variability in the evaluation data. Experimental results presented on a data set of 50 Arabic documents, handwritten by different writers, demonstrated that the subspace method achieves approximately 88% for both recall and precision on the 50 test images.

2.2 Image Enhancement [Polar Rain, SUNY]

Shape-DNA based Image Enhancement [Polar Rain]: In this phase, we performed experiments to assess the usefulness of shape-DNA enhancement on machine-print and handwritten images. The shape-DNA approach uses a database of low- and high-resolution shapes and a probabilistic shape-mapping model. The database and mapping are both automatically learned from training data to estimate high-resolution details from low-resolution shapes using shape-based computation models.

For our initial pre-processing experiments we created a corpus of images from two different sources: (1) 300 real-world documents acquired primarily in Afghanistan, and (2) a controlled document set of 33 images generated by three different Iraqi writers each writing the same 11 distinct documents. Using this corpus of images, two sets of preliminary experiments were performed to separately assess the restoration and cleaning attributes of the shape-DNA approach. In the first set of experiments the usefulness of the restoration was assessed. We first estimated shape databases using 15 of the 33 controlled handwritten documents. Then, we synthetically degraded the remaining 18 controlled documents by introducing breaks in character glyphs and degradations that mimic those observed in real-world documents. Finally, we applied the shape-DNA technique to restore the broken characters. In the second set of experiments, our goal was to evaluate the cleaning of real-world documents *without* restoring the text. For these experiments we did not train on the controlled handwritten data. Instead we used shape databases trained on machine-printed Arabic documents. We assessed the effectiveness of the restoration and cleaning processes through a visual comparison of the output with the input image. The comparison showed that the shape-DNA technique is capable of cleaning age-related degradations and smudge marks in real-world documents. On the controlled set of handwritten documents the shape-DNA technique was able to recover most of the missing pieces of the broken characters.

We further focused on configuring shape-DNA enhancement on documents in a DIA-supplied corpus of Arabic text as well on documents from the MADCAT handwritten corpus. The images in the DIA corpus are very noisy and also contain a fair number of non-text objects, with handwritten text often overlaid on these non-text objects. Therefore, it is important to detect text in such documents before cleaning or restoration. We also observed that the projection of text-only segments onto shape-DNA models results in much smaller distances compared to the projection of non-text segments onto the same models. Based on this observation, we explored a novel application of the shape-DNA approach, namely, to differentiate text-only segments from non-text segments. For our initial experiments, we used shape-DNA models trained on printed text documents. Next, we used these models to detect text-only regions in samples documents from the DIA corpus. Preliminary results indicate that this approach is a viable one for text detection and verification. As shown in Figure 3, our approach is robust to segments that contain

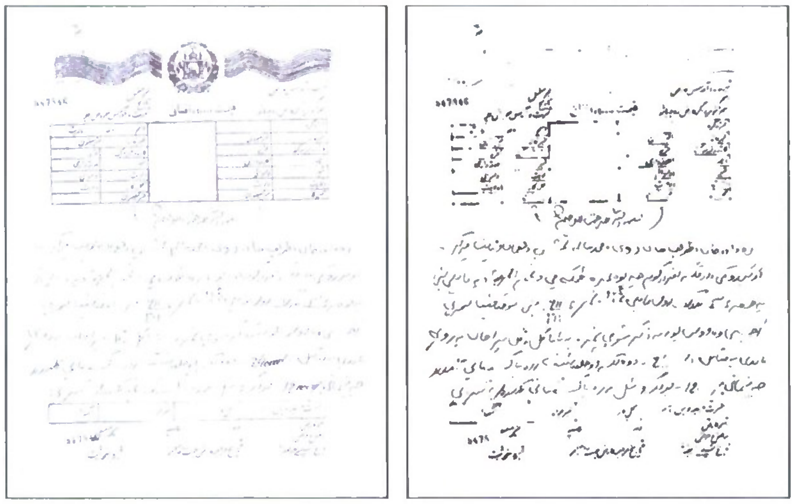
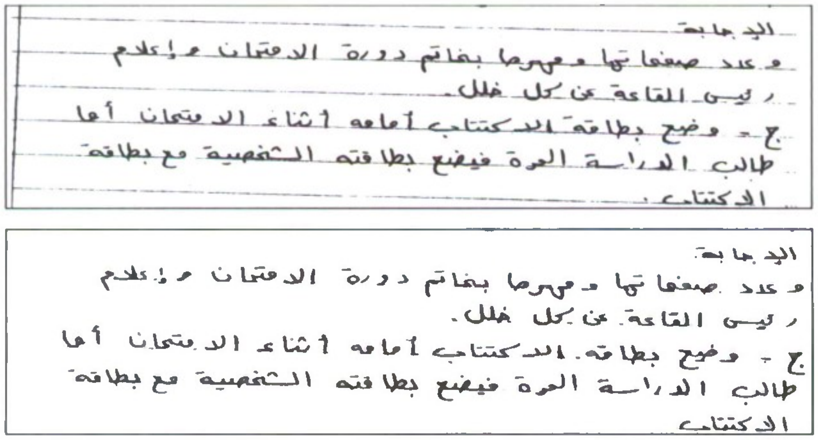


Figure 3: Shape-DNA on DIA sample from Afghanistan.



الإجابة
و عدد صفحاتها وختمها بخاتم دورة الامتحان وإعلام
رئيس القاعة عن كل خلل.
ج - وضع بطاقة الاكتتاب أمامه أثناء الامتحان أما
طالب الدراسة الحرة فيضع بطاقته الشخصية مع بطاقة
الاكتتاب.

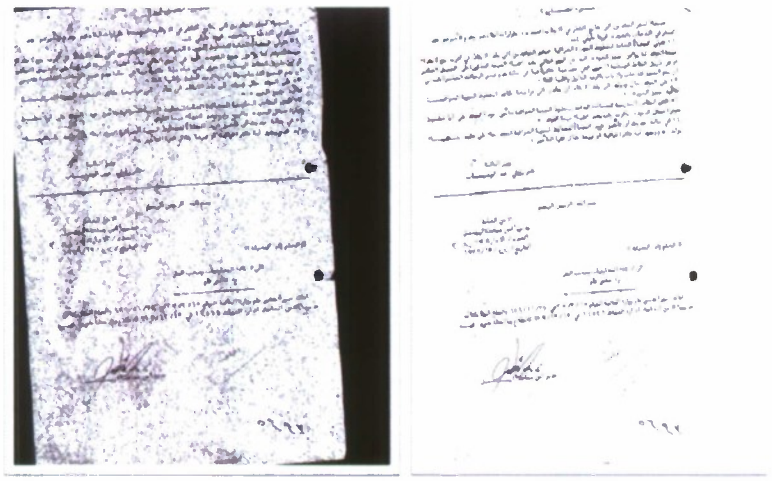
الإجابة
و عدد صفحاتها وختمها بخاتم دورة الامتحان وإعلام
رئيس القاعة عن كل خلل.
ج - وضع بطاقة الاكتتاب أمامه أثناء الامتحان أما
طالب الدراسة الحرة فيضع بطاقته الشخصية مع بطاقة
الاكتتاب.

Figure 4: Shape-DNA on AMA sample.

a mix of text and non-text objeet.

We also applied shape-DNA enhaneement to images in the Applied Media Analyties (AMA) eolleeted handwritten (HW) eorpus. Sinee the AMA images eontain artifaets sueh as page lines, extraneous dots, seanning artifaets, ete., we eustomized the restoration to handle sueh degradations. Figure 4 shows a sample output for the shape-DNA restoration applied to an AMA sample eontaining page lines and other seanning artifaets.

Morphology-based Image Enhancement [SUNY]: In this phase, we additionally developed a noise removal and image enhaneement algorithm based on a eombination of morphology-based and the region growing binary image enhaneement algorithm. An example is shown in Figure 5.



Degraded Image **Enhanced Image**

Figure 5: Example of Morphology-based Image Enhancement.

3. Accomplishments in Page Segmentation [Argon, Lehigh, Polar Rain, SUNY, UMD]

The objeetive of the page segmentation task is to deeompose an image into "pereeptually eonsistent" zones, where pereeptual eonsisteney is deseribed in terms of loeal texture similarities, with the goal of produeing zones that eontain a single type of eontent (maehine-printed text, handwritten text, graphies, etc.). In addition, aeeurate segmentation of text zones into lines of text is critieal for doeument level OCR systems. The MADCAT dataset distributed by LDC contains many doeuments where the handwritten text is very erowded and there is skew in multiple direetions within the same doeument. Grouping small elements sueh as dots and other diaeritics with the right text line is a big ehallenge for handwritten text. In this phase we foeused on segmenting the image into pereeptually eonsistent zones *and* line segmentation for text zones.

3.1 Page Zoning [Lehigh, UMD]

Pixel Level Zoning [Lehigh]: During this phase, we investigated pixel-based approaehes for page segmentation that, while eomputationally demanding, offer the potential to be general and robust. Our teehniques employ a range of elassifier teehnologies, ineluding brute-foree k-Nearest Neighbors (kNN), fast approximate kNN using hashed k-d trees, elassifieation and regression trees, and loeality-sensitive hashing. Initial experiments suggested that per-pixel aeeuraeies were modest, in the range of 60% to 70%. Aeeuraey improvements of up to 80% were aehieved by refinements in ground-truthing protocols. Recent algorithmic improvements to our approximate kNN classifier using hashed k-D trees allows trade-offs between runtime and accuracy which are highly promising: a 23-times speedup with less than 0.1% loss of accuracy; or a 60-times speedup with less than 5% loss of accuracy. We plan to investigate this further in the next phase.

Voronoi-Tessellation Zoning [UMD]: Page segmentation techniques typically divide the image into non-overlapping rectangular boxes around perceptually consistent zones (text, graphics, tables, etc.). But in pages where the layout is arbitrary, extracting non-overlapping rectangular zones is not only infeasible but also undesired. We have therefore developed a page segmentation algorithm based on Voronoi-tessellation to extract arbitrary-shaped zones with homogeneous content. The algorithm proceeds as follows:

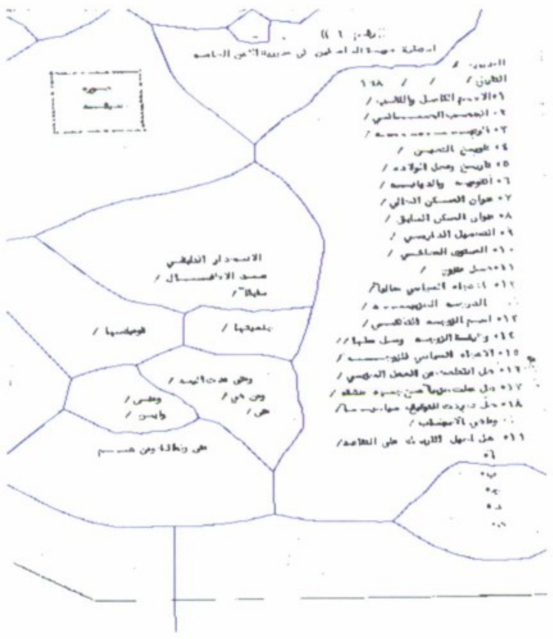


Figure 6: Example of page zoning using Voronoi-Tessellation algorithm

1. A page is segmented into regions using Voronoi tessellation – based on the connected components to obtain the candidates of boundaries of document components.
2. The candidates are utilized to estimate the inter-character, inter-line and inter-column gaps without the use of domain specific parameters so as to select the boundaries.
3. The regions are separated by a series of small line segments marking boundaries between adjacent regions.
4. Line segments surrounding each region are sewn together to form a polygon. Hence, polygonal-based region segmentation is achieved.
5. Each region-polygon is approximated to contain minimum number of edges.

Our initial assessments shows that the basic Voronoi cell approach performs reasonably well on clean documents, but noise artifacts result in creation of false regions. An example of the performance of the algorithm is shown in Figure 6.

3.2 Text Line Segmentation [Argon, Polar Rain, SUNY, UMD]

Local Concavity Map based Text Line Segmentation [SUNY]: We implemented a line finding library using an adaptive local connectivity map. In this approach, we generated a location mask for the line pattern and extracted text lines by superimposing the location mask on the original image. While this works reasonably well for most pages, we found that the performance degrades on documents with crowded lines and wide variability in skew. Therefore, we implemented a new line-segmentation algorithm using a steerable filter to assist the text line segmentation process. The steerable directional filter is a mathematical image filter that finds the local orientation of a text line by scanning in multiple directions. The maximum response from a convolution of the filter with the image gives the text line direction. Finally, the image is modified into a text line pattern map.

Another problem in text line segmentation is the problem of separating components that span adjacent lines or are touching components from other lines. In order to tackle this problem, a stroke segment that crosses the text line is automatically detected and annotated. A splitting reference line is found using center of gravities of the contours and splitting is done only for the detected touching pieces. The splitting is done at the contour level which allows a reconstruction of the character images. Figure 7 shows a crowded page and the result from the text line finding module. The highlighted areas are bounding boxes of touching characters crossing different text lines. The split characters are reconstructed and accurately grouped with the text lines that they belong to.

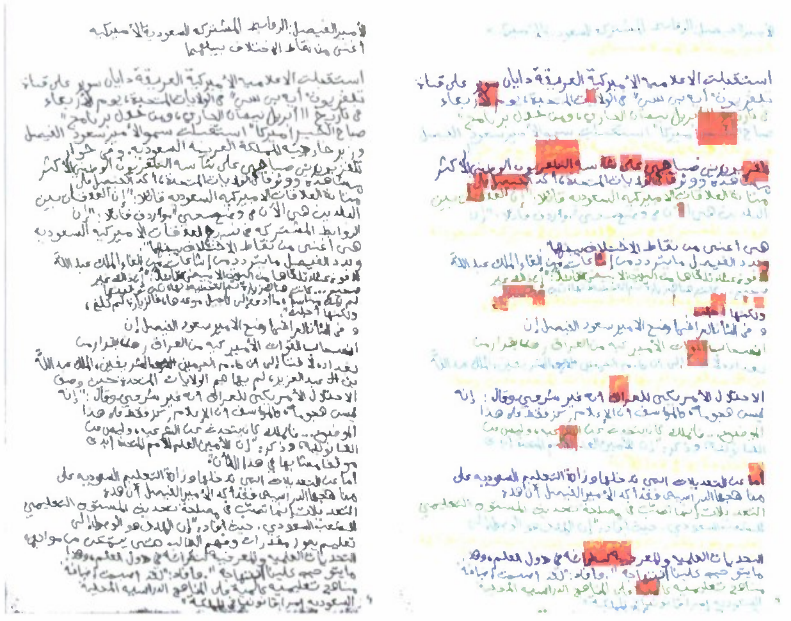


Figure 7: Example of Text-line separation using Steerable filter

Affinity Propagation based Handwritten Line Segmentation [UMD]: To overcome the challenges in line detection for handwritten Arabic, we developed a multi-stage approach to the task. In the first stage, we automatically determine an area threshold that is used to remove diacritics. Next, we fit a 2-Gaussian mixture model to the areas of the connected components. The Gaussian component with the smaller mean value represents the average diacritic area and the Gaussian component with the larger value represents the average non-diacritic area. In order to determine document zones with different orientations, we apply the Affinity Propagation (AP) method. AP is an unsupervised clustering method that depends on computing a pair-wise similarity measure between pairs of data points. To detect text lines within each text zone, we reapply AP on individual zones. However, here we compute the pair-wise similarity based on the vertical location of the connected components. In the next phase, we will explore tuning of the parameters to optimize line segmentation accuracy.

Baseline Detection and Slant Correction [Argon]: We implemented a technique for detecting and correcting curved and slanted baselines. First, we skeletonized and filtered connected components not associated with the baseline. Next, a sliding window encompassing at least three baseline related components was used to get sufficient context for each sub-word. Finally, local minima and junction points were detected, and Random Sample Consensus (RANSAC) algorithm was used to fit a baseline to the original skeleton. For slant correction, each endpoint above the baseline was traced back to the vicinity of the baseline, and RANSAC was again used to fit a straight line to the path. If more than 50% of the points on this line are returned as inliers by RANSAC, then it is considered sufficiently straight to use for slant correction. We have not investigated what effect, if any, slant correction has on our recognition rate. A more formal evaluation is currently planned for Phase 2.

Text Verification [Polar Rain]: We developed a fast and robust text verification algorithm based on Shape-DNA models. This algorithm can be used either as a tool for text segmentation, or for identifying homogeneous image regions and for detecting whether homogeneous regions of interest are non-text or text regions. The algorithm can also be used to identify the text type, i.e., printed or handwritten. In our approach, we first project input image segment onto a database of Shape-DNA patterns. Next, we computed histogram statistics of projection distances. Histogram of projection distances for text and non-text images as well as printed and handwritten text have different characteristics, and this feature is exploited in classifying the text images. Our approach takes only 0.5 sec for processing a 1200x800 pixel image.

Texture-based Text Detection in Camera-captured Images [Argon]: Our approach for text detection in camera-captured images is based on measuring shape similarity of texture. However, this technique is not suitable to natural images. In this phase, we developed an initial version of the text detection module using texture-analysis based approaches. We used different sets of features including wavelet coefficient histograms, color, edge orientation, and shape features. These were divided into three main groups (wavelet, color, edge/shape) and a Support Vector Machine (SVM) classifier was trained on each. The results of these SVMs were combined using AdaBoost with logistic sigmoid regression. For training, we have implemented a graphical user interface to allow for manual construction of ground-truth data. The tool ingests an image, partitions it into super pixels (collection of similar pixels), and classifies each super pixel. These classifications can then be manually corrected, and the corrections fed back in as new examples to train the SVM classifier. In the coming months, we will run experiments to measure accuracy of the text detection module. We trained the SVM and AdaBoost classifiers for text detection on an internally collected 29 American-English camera-captured images of signs. We also implemented a previously published Plane Parameter Markov Random Field (PP-MRF) approach for text rectification. Initial assessment of this approach suggests that significant improvements are required to improve performance.

4. Accomplishments in Text Recognition [BBN, SUNY, Argon, Columbia]

The goal of text recognition is to digitize the text in image zones that have been determined as containing text. In Phase 1 of the effort, we made several advancements in our hidden Markov model (HMM) text recognition framework. Additionally, we explored matching using structural features such as graph elements and a novel framework for integrating HMM-based and structural matching techniques.

In this section, we first provide an overview of BBN's Hidden Markov Model (HMM) based text recognition. Next, we present a suite of techniques for improving HMM-based text recognition and present experimental results on the DARPA Arabic machine printed corpus as well as the Phase 1 MADCAT handwritten corpus released by the Linguistic Data Consortium. We conclude with results from a novel integration of HMM based recognition and structural matching.

4.1 Improvements in HMM based Text Recognition [BBN]

4.1.1 Overview of HMM-based Text Recognition

Figure 8 shows a block diagram of our HMM-based text recognition system. The images are first pre-processed to remove any skew due to rotation of the text during scanning. They are then segmented into lines of text using an HMM-based line finding algorithm. For each line of text, features are computed from a sequence of overlapped windows (also called frames). The feature extraction program typically computes several features for a single frame. A total of 33 of the following script-independent features are extracted from each frame: Percentiles of intensity values, Angle, Correlation, and Energy (PACE). Linear Discriminant Analysis (LDA) was then applied to reduce the dimension of the feature space from 33 to 15. The resulting vector of 15 LDA features is a compact numerical representation of the data in the frame, and is the feature vector used for training and recognition.

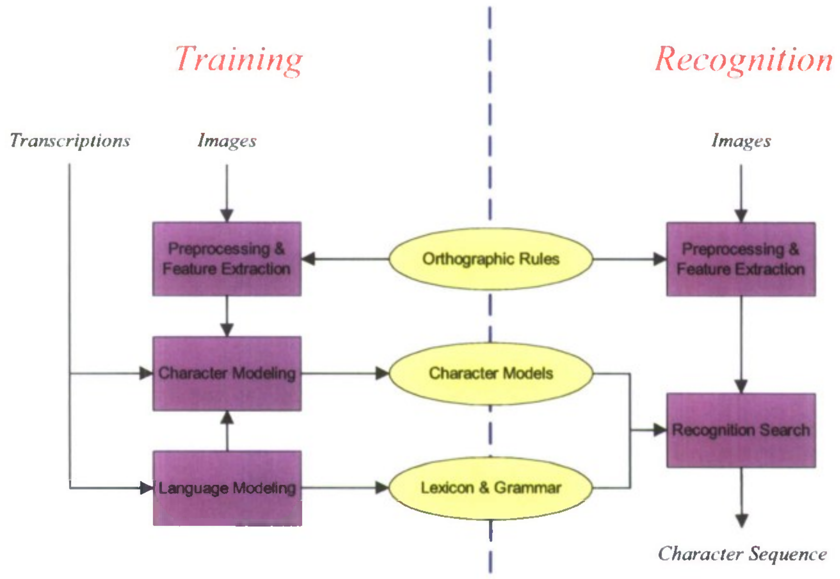


Figure 8: Block diagram of the BBN OCR system.

We use a 14-state left-to-right Hidden Markov Model (HMM) to model each character or glyph. The model for a word is the concatenation of the HMMs for the individual characters in the word. Each state of the HMM has an associated output probability distribution over the features which is modeled as a weighted sum of Gaussians. The estimation of the parameters of the HMM is performed using the Expectation Maximization (EM) algorithm. In order to reduce the number of parameters to be estimated by the EM algorithm, we share Gaussians across different character models.

Arabic is a connected script where each character can be rendered in a different graphical form based on its position within a word. However, the underlying sequence of characters in the ground truth transcripts is a sequence of Unicode characters. These Unicode characters are inherently *context independent* and typically the browser or the editor renders the character in the "presentation-form" based on the neighboring characters. We refer to the context independent Unicode character as the "base-form" representation. For training glyph HMMs, we first apply a set of transformation rules to convert the base-form character transcripts into presentation-form transcripts. Next, we estimate a separate HMM for each presentation-form character. Training on the contextual form of the Arabic character rather than the base form of the character captures the change in a character glyph due to the neighboring characters.

The decoding process is a two-pass beam search for the most likely sequence of characters given the observed feature vectors. The forward pass is an approximate but efficient procedure for generating a small list of character sequences that are possible candidates for being the most likely sequence. The backward pass is a more detailed search for the most likely character sequence within this small list.

4.1.2 Baseline Experiments on Machine-printed Images

In Phase 1 of the MADCAT effort, prior to the release of the handwritten data, we performed our text recognition research on the 1995 DARPA Arabic machine-print (DAMP) corpus collected by SAIC. The corpus consists of 297 images scanned from newspapers, books, magazines, etc. The corpus was partitioned into three sets: 60 images for development, 60 images for testing, and 177 images for training the OCR system. In addition to the 177 images from the DAMP corpus, we used 380 synthetically generated images. These 380 images were created by printing 100 newswire text passages in multiple font types and font sizes.

In our experiments on the DAMP corpus, we used 14-state context-independent HMMs for modeling the contextual form of Arabic characters and a word or character n-gram language model (LM). From the training data, we estimated HMMs for a set of 162 presentation-form characters. This set includes Arabic characters, Arabic numerals, and some additional non-Arabic numerals and characters. A total of 68K Gaussian mixtures were estimated with a maximum of 512 Gaussian components assigned to each character in the training lexicon.

Next, we estimated both character and word LMs from transcriptions of the images from the training set as well as from 2.6 million words of Arabic newswire data. The character lexicon used for character *n*-grams consisted of all 162 characters observed in the training images. The word lexicon was restricted to the 65K most frequent words in the LM training data. Both the character and word LMs were trained using Witten-Bell discounting.

We performed recognition experiments on the test set to compare character LM to word LM. The configuration used for recognition has two steps. First, the two pass decoding described above was used to generate an N-best list (character or word N-best depending on the type of language model). Next, we rescored the N-best list using a trigram word or character LM). The weights for combining different knowledge sources in the N-best rescoring were estimated on the development set. In Table 1, we summarize the word error rate (WER) for both the word and character LM measured on the test set. As shown in the table, the character trigram LM resulted in a WER of 12.3% compared to 15.9% obtained by using a word LM. Although a word trigram models wider context than a character trigram, in our recognition experiments the WER for the word trigram is higher than the character trigram LM. We believe this is because of the high out-of-vocabulary (OOV) rate of the word LM. The word lexicon results in an OOV rate of 12.6% on the test set, therefore the errors are dominated by OOVs.

Language Model – Trigram	%WER
Word	15.9
Character	12.3
PAW	10.1

Table 1: Comparison of different LMs for recognition on the DAMP test set.

4.1.3 Improvements on Machine-Printed Images

We explored techniques for improving the language model (LM) as well as the glyph models on the DAMP corpus.

Higher-order LMs: For the same *n*-gram order, a character *n*-gram LM models a much narrower context than the word LM. Therefore, to model a wider *n*-gram context using characters, we estimated *n*-gram character LMs with $n > 3$. Since our two-pass decoder only supports LMs with $n <= 3$, we used the higher order LMs only for rescoring the N-best list generated by decoding with a trigram character LM. As shown in Table, 2 increasing the n-gram order to five decreased the WER by 0.8% absolute over using the trigram LM. However, increasing the n-gram order beyond five did not yield additional WER reduction.

PAW-based LMs: A Part-of-Arabic word (PAW) is a character sequence that is typically rendered as a single connected component in an image. PAWs can be sub-words or words and are derived from the morpho-lexical rules of the Arabic language. An LM trained with PAWs as lexical units is likely to provide wider context at the same n-gram order than a character LM, while still preserving the unlimited vocabulary coverage property of the character LM.

Character N-gram Order	%WER
n=3	12.3
n=5	11.5
n=7	11.7

Table 2: N-best rescoring experiments with character LMs with n>3.

We estimated a PAW trigram LM from the same training data used for character and word LMs. A total of 9K PAWs including individual characters were used as lexicon units in the PAW LM. The number of PAWs was restricted to 9K by imposing a length cutoff of six characters for the PAWs. The length cutoff was empirically determined on the development set by decoding with different cutoffs. Next, we decoded the test set using the PAW LM and the glyph HMMs described in Section 2.3.2.1. Finally, the N-best was rescored using the PAW trigram LM. As shown in the last row of Table 1, the PAW trigram LM resulted in a WER of 10.1%, a significantly lower WER than the word and the character trigram LM. We also explored using a higher order PAW LM for N-best rescoring. However, experimentations with *n*-gram order > 3 using a PAW LM did not show any additional improvement.

Context-dependent Glyph HMMs: In Arabic script the shape of a character glyph often varies based on the position of the character within a word. One approach for modeling this context dependency is to use presentation-form characters as modeling units for training context independent (CI) HMMs. Another alternative is to use base-forms to model glyphs, but instead of using CI HMMs, use context-dependent (CD) HMMs.

We performed context-dependent (CD) training with both presentation form characters and base form characters. We also used two different configurations for tying HMM parameters. In the first configuration, referred to as character-tied mixture (CTM), a single mixture of Gaussians is shared by all contexts of a particular character. In the second configuration, which we refer to as position-dependent tied mixture (PDTM), a separate set of Gaussians is estimated for each state of *all* the context-dependent HMMs associated with a particular character. For example, we estimate a set of '*K*' Gaussians for the first state of all HMMs associated with the character "Alif", and a separate set of '*K*' Gaussians for the second state of all HMMs for "Alif", and so forth.

In addition to sharing the Gaussians, we used a decision-tree based clustering of mixture weights for both the CTM and PDTM configuration. The decision-tree uses a set of questions based on different characteristics of the characters, e.g. whether the character is an ascender or a descender.

In Table 3, we compare the performance of different CD configurations with the baseline configuration described in Section 4.1.3. For fair comparison all models were configured to have approximately the same total number of Gaussian mixtures as the baseline configuration.

For comparing the different CD models, we decoded the test set using the word LM described in Section 4.1.2. As shown in Table 3, CD training using presentation form characters as modeling units did not yield any improvement over the baseline CI configuration. A possible reason for this result is that the contextual form characters, by definition, model glyph variations depending on the relative position of the character within a word. Thus, any additional attempt at modeling context merely fragments the training data.

Unlike presentation form characters, context-dependent modeling using base forms characters with PDTM tying resulted in a 0.7% absolute reduction in WER over the baseline result of 15.9% obtained using CI modeling with presentation form characters.

Training Configuration	**%WER**
Pres. form, CI (baseline)	15.9
Pres. form, CD, CTM	16.9
Pres. form, CD, PDTM	16.9
Base form, CD, CTM	16.2
Base form, CD, PDTM	15.2
+ MCE training	14.1

Table 3: Decoding the test set with context-dependent HMMs and word LM.

Discriminative Training: Traditionally, HMM parameters are estimated using ML criterion. However, in recent years discriminative training has been shown to outperform phonetic HMMs estimated using ML for speech recognition. Standard ML estimation attempts to find model parameters that maximize the likelihood of the training data. In contrast, discriminative training attempts to make the correct hypothesis more probable while simultaneously making incorrect hypotheses less probable.

Similar to Minimum Phone Error (MPE) used in speech, recognition, we define the following objective function to maximize for performing *Minimum Character Error* (MCE) training for glyph HMMs:

$$F_{MCE}(\lambda) = \sum_{i=1}^{N} \log \frac{\sum_h P_\lambda(O_i | M_h) P(h) CharAccuracy(h)}{\sum_h P_\lambda(O_i | M_h) P(h)}$$

In the equation above, the *CharAccuracy(h)* is a measure of the number of characters accurately generated in hypothesis h, O_i are the feature vectors, λ are the HMM parameters, is $P_\lambda(O_i|M_h)$is the glyph model score, and $P(h)$ is the LM probability.

Next, we used character lattices generated using ML glyph models and a unigram character LM to perform MCE training with base form CD HMMs with PDTM parameter tying. The extended Baum-Welch algorithm was used for updating parameters and I-smoothing was applied to avoid over-training. As shown in Table 3, decoding the test set with the MCE models and the word trigram LM resulted in a WER of 14.1%, a 1.1% absolute reduction in WER over the ML models.

Finally, to make use of the best glyph HMM and the best LM, we implemented the following *hybrid* recognition setup. First, we used the CI presentation form glyph HMMs estimated using ML and the PAW LM to generate an N-best list. Next, the PAW N-best was converted into a word N-best. Finally, the base form CD PDTM glyph models estimated using MCE was used to rescore the word N-best list.

Recognition Configuration	%WER
Decoding + N-best rescoring with word LM and ML CI pres. form HMMs	15.9
Decoding with PAW LM and ML CI pres. form HMMs + N-best rescoring with CD PDTM base form HMMs	9.6

Table 4: Summary of improvements in WER using a hybrid decoding and rescoring strategy.

In Table 4, we compare the results from the hybrid recognition configuration above to the baseline recognition configuration with ML CI glyph HMMs and word LM. As shown in the table, we have achieved a 40% relative reduction in WER over our baseline configuration.

4.1.4 Baseline Experiments on MADCAT Handwritten Corpus

During the 4-month period June-December 2008, LDC released a total of 9741 scanned images of handwritten Arabic text of newswire articles, weblog posts, and newsgroup posts, along with the corresponding ground truth annotations. First, we trained glyph HMMs after each release using the Percentile, angle, correlation, and energy (PACE) features used as baseline feature set. We also measured the effect of incrementally increasing the amount of training data used for glyph modeling on text recognition performance.

Feature Extraction: In order to convert the 2-dimensional images into a 1-dimensional sequence of features needed to build HMMs, we typically determine the location of the top and bottom boundaries of the lines of text, and then compute the feature vector for each of these lines. If the lines are regularly spaced as in machine-printed data, HMM-based or connected component based line-finding algorithms achieve near-perfect accuracy. However, in free-flowing handwritten data, adjacent lines of text often overlap. Also, handwritten text exhibits significant variations in baseline within a text line. This is an issue for feature extraction since the width of the frame is set proportional to the height of the line. All of the above factors make text line finding/separation for handwritten text a difficult problem. For our experiments in this phase, we used the rectangular bounding boxes on the individual word images to obtain a piece-wise linear approximation of the envelope for the text line. Features were computed from the piece-wise linear envelope around the line image. The left and right boundaries of the word were not used for feature extraction. The PACE features were used as the baseline feature set.

Training: In handwritten Arabic text, the shape of the character glyph often varies depending on the characters that precede and follow it. Such context-dependence of glyphs is typical of cursive connected scripts, but can vary even more widely because of a writer's personal style. Context-dependent HMMs offer a robust, data driven approach for modeling contextual information. We found that context-dependent models out-performed over context-independent models for machine-printed Arabic text. We trained Position-dependent tied mixture (PDTM) HMM models, where a separate set of Gaussians is estimated for each state of all the context-dependent HMMs associated with a particular character. In total 254K Gaussians were trained for 176 unique characters (including Arabic characters, numerals, punctuations and English characters).

Recognition: A trigram language model trained on 90 million words of Arabic newswire data was used for recognition. The decoding lexicon consisted of 92,000 of the most frequent words in our Arabic text corpus. The out of vocabulary (OOV) rate of the test set measured against the 92K lexicon is 7.5%. The forward pass is a fast match beam search using the HMMs and an approximate bi-gram language model. The output of the forward pass consists of the most likely word-ends per frame. The backward pass operates on the set of choices from the forward pass to restrict the search space and an approximate trigram language model to produce an N-best list of hypotheses. The N-best list is then re-ranked using a combination of the acoustic scores, and a language model score. The weights for re-ranking were tuned on the development set.

A separate set of 442 images released by LDC was split into two parts; one was used for development and the other for testing. Table 5 shows the performance on the test set with varying amounts of training data for glyph modeling. Increasing the amount of glyph model training data by a factor of 3 resulted in an 8.7% relative drop in WER for authors who were never seen in training. Note that the WER was measured by detaching punctuations and sequences of digits from other words to which they may be attached. Surprisingly, the WER on the images by authors in training is much worse than those by authors who are not in training. We believe that this is an artifact of the data and scribe selection of the test set under consideration. Experiments on a different test set held out from the training data showed the performance on scribes not seen in training to be 31% relative worse than those who were represented in training. Note that increasing the amount of data improves performance on the pages written by authors who are not seen in training. However, this is not the case for authors represented well in training.

Number of Training Images	Number of Training Authors	%WER	
		Authors in Training	Authors not in Training
848	10	51.3	36.3
3371	20	41.1	31.1
5288	38	41.6	29.4
8253	58	43.8	28.6
9741	71	43.1	28.4

Table 5: OCR Performance with Glyph models trained on different amounts of training data

4.1.5 Improvements on Handwritten Corpus

In the following we summarize the improvements in HMM-based text recognition on the MADCAT data.

Unsupervised Adaptation [BBN]: Adaptation has been widely used to combat the variability in speech in automatic speech recognition and to adapt to fonts and degradations in text recognition of machine-printed documents. In handwritten text variability occurs due to inter-scribe differences in writing style, font and slant.

We performed MLLR estimation to update the Gaussian means of the HMMs using the text recognition output of each page in the test set. The updated models were then used to re-decode the given page. Unsupervised adaptation gave a relative improvement of 4.5% as shown in Table 6.

Improvements	%WER
PACE Features (Baseline)	35.7
+ Unsupervised Adaptation	34.1
+ Gradient & Concavity Features	31.3

Table 6: Summary of handwritten text recognition improvements on MADCATdata.

Integration of Structural Features in HMM based Glyph Modeling [BBN, SUNY]: In this phase, we explored two new *structural* features for text recognition – Directional Element Features (DEF) and Gradient-Structure-Concavity (GSC) features. The directional element features are based on the idea of non-linear matching of the directions of the patterns, and are variants of the gradient features. In the Arabic script, many letters share common primary shapes and differ only in the number and position of the dots and strokes. Structural features capture intuitive aspects of writing such as loops, branch-points, endpoints, and dots. One such family of features is the GSC (Gradient, Structure and Concavity) features. GSC features are symbolic, multi-resolution features that combine three different attributes of the shape of a character – the gradient representing the local orientation of strokes; structural features that extend the gradient to longer distances and provide information about stroke trajectories; and concavity that captures stroke relationships at long distances. While GSC features have successfully been used in recognition of isolated digits and handwritten words in the past using segmentation-based approaches, they have never been used in the HMM framework. In this section, we describe a novel integration of these structural features in BBN's HMM-based text recognition framework.

For computing the GSC features, first a gradient map is constructed from the normalized image by estimating gradient value and direction at each pixel. Next, Gradient features are obtained by counting the pixels which have almost the same gradient. The structure features enumerate complex patterns of the contour. To compute the concavity features, pixels which lie in certain special regions such as holes and strokes are detected. The image is then divided into bins and the number of such pixels in each bin is counted.

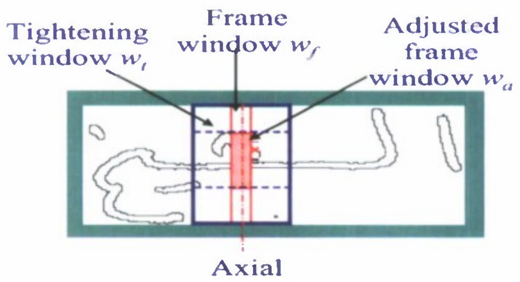


Figure 9. Illustration of frame tightening for feature computation.

In Phase 1 of the effort, we integrated GSC features into the HMM framework as follows. First, we used a wider window than the PACE features for computing the GSC features. Since the baseline of a word image may fluctuate within portions of the same word, we also algorithmically tightened the upper and lower boundaries of the sliding window. This ensures that the features are normalized and also minimizes the variation of the feature space. Figure 9 shows thematically the tightening of the frame. We define the width of the frame (w_f) as w_f.width = w_f.height/12, where w_f.height is the height of the word image. A tightened window wt is obtained by expanding w_f to both the left and right sides so that w_t.width = 5w_f.width. The upper and lower boundaries of w_f are redefined by the bounding box of black pixels within w_t. We thus obtain an adjusted window w_a. The region within w_f which is outside of w_a does not have any black pixels. The GSC features are computed from the adjusted frame window w_a. The tightened window w_a is divided evenly into 12 overlapping vertical bins, and 4 sets of GSC features are computed for each bin. A total of 48 of each of the GSC features are computed for each frame, and we use an LDA to reduce the dimension of the feature vector to 15.

We tried different combinations of the PACE, DEF and individual GSC features with and without frame tightening. Frame tightening consistently showed improved performance. The DEFs provided a 4% relative improvement in WER. The combination of Gradient and Concavity features with the PACE features yielded the biggest gain – an 8% relative reduction in WER as shown in Table 6.

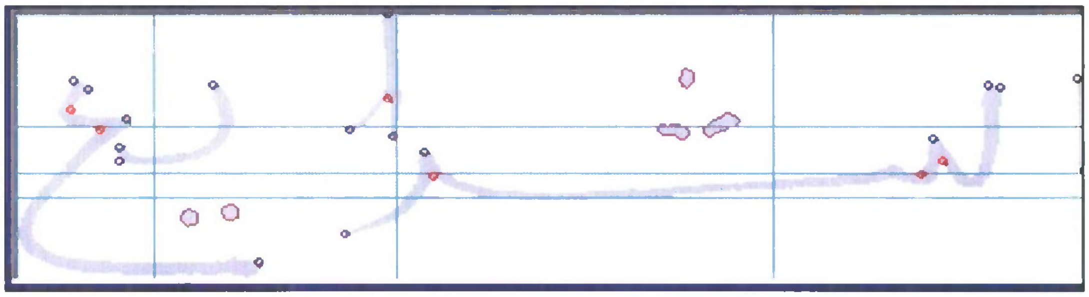


Figure 10: Example of critical point and chain code features extracted by SUNY.

Chain-code and critical point Features [SUNY]: In addition to the integration of GSC features into the HMM framework performed by BBN, our team investigated other novel features for improving handwriting recognition. In particular, SUNY developed the critical point chaincode features. These features trace the "chaincode" of text and identify critical points such as sharp turning points, end points, crossing points, and connection points in the text. Critical points are ordered in the sequence that they are encountered during the trace and serve as a coded feature vector. Figure 10 shows a sample output from the feature extraction library for chaincode and. In the next phase, we plan to integrate these features in the HMM-based text recognition system.

Slant Correction [BBN]: In order to normalize the hand-written documents by different authors, we pre-processed the images by automatically correcting the slant in each word image by measuring the relative pixel organization along the perimeter of connected components and then used these statistics to reorganize each pixel position to reduce the overall slant in the image. The slant-corrected images used for text recognition did not result in any improvements.

Page-line Removal Experiments [BBN]: It was found that the performance of the text recognition system was considerably worse on pages with horizontal rules compared to pages without such rules. We pre-processed the images to remove lines using three line-removal algorithms from: a) Polar Rain, b) SUNY, and c) BBN- Heuristic. The details of the algorithms are described in Section 2.1. The results in Table 7 on the test set pages containing ruled lines compare the three algorithms against a baseline without any line removal. The SUNY and BBN algorithms result in a small improvement in performance over the baseline, with SUNY's technique giving a relative improvement of 1.8%.

Page-line Removal Algorithm	% WER
None	38.1
SUNY	37.4
BBN	37.9
Polar Rain	39.3

Table 7: Comparison of performance of different line removal algorithms on test set images with ruled lines.

In Figure 11, we show an instance of line-removal performed by the three algorithms on a section of an image. The SUNY algorithm successfully removes lines from the image, but doesn't perform noise-removal in addition to line-removal as done by the BBN and Polar Rain algorithms. The BBN algorithm removes some character glyphs which are connected to the line while performing line removal resulting in disconnected character segments in the image. Since the feature extraction algorithm does not rely on performing connected component analysis, the creation of disconnected components does not significantly affect system performance. The Polar Rain algorithm also removes some text pixels associated with the character while performing line removal.

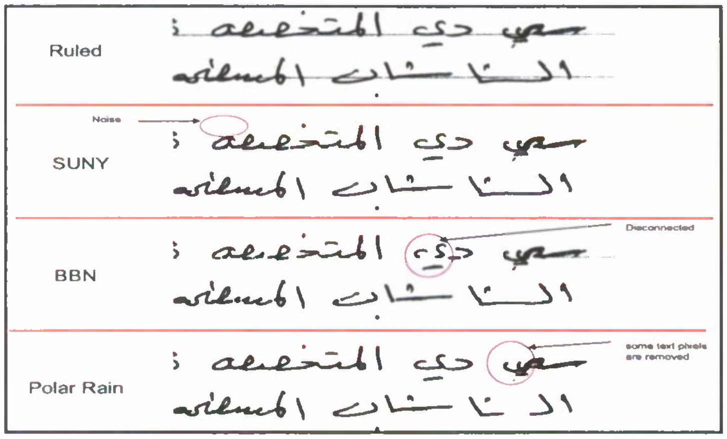


Figure 11: Examples of line removal using the different algorithms.

Additionally, we also trained our HMM-based text recognition system with MRF ruled-line removal and restoration algorithm applied to all images. Next, we decoded the test set with the models trained using GC+PACE features. Using the MRF ruled-line removal resulted in a modest improvement of 0.6% in the WER.

Improved Language Modeling [Columbia, BBN]: In this phase, we developed several different n-gram language models, based on different Arabic word representations. Orthographically defined representations included the basic word forms, the word-parts (sequences of graphically connected sub-words), letters (graphemic stand-alone form), and letter shapes (i.e. glyph-like representations that remember the contextual allographic form of the letter). We also explored versions of these models that remove all dots, since missing and misplaced dots are common handwriting errors. Morphologically defined representations included diacritized and undiacritized lexemes and basic POS tags. We also began exploring more complex Factored Language Models, which provide a means to combine several features into one model. The models were trained using various volumes of data taken from the Arabic Gigaword corpus. The lexeme and other morphological models required the use of MADA to perform lemmatization as an initial step before training the model. We integrated the initial version of Columbia's LM rescoring module into BBN's text recognition system and are presently optimizing it on the MADCAT data.

4.2 Probabilistic Bipartite-Graph Matching [Argon]

Graph-based features: In this phase, we attempted to capture the structural characteristics of text by extracting graph-analytic features. The feature types considered were loops, dots, endpoints, corners, 3-ways, 4-ways, and nodes of degree greater than four. Figure 12 shows examples of some graph features in Arabic.

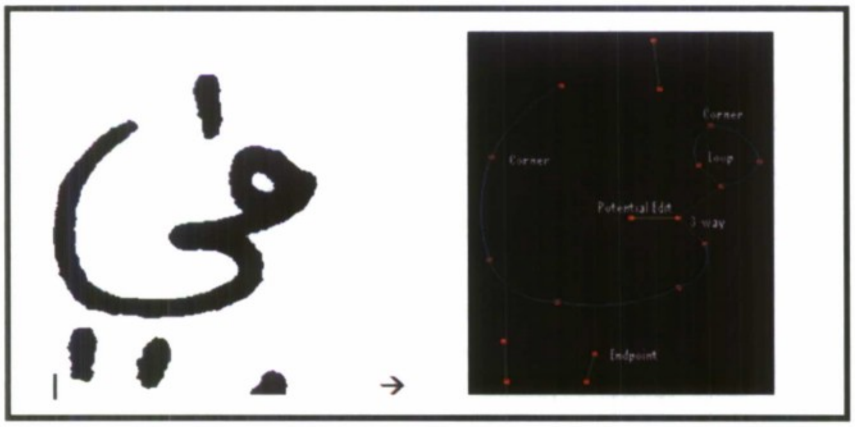


Figure 12: Example of Graph features in Arabic

The generation of these graph features consists of two main steps:

1. Graph Generation: First, we apply a thinning algorithm to reduce text glyphs to 1-pixel wide skeletons. Next, we convert the skeleton to a graph representation by identifying intersections, end-points, and loops.
2. Feature Computation: We compute discrete structural symbols such as arcs, loops, intersections, etc. or, extract continuous attributes of above symbols including relative position, orientation, and angle between edges.

Graph Matching: In the document understanding seedling effort, we performed bipartite matching against selected features to generate pairings between features in the graph. In the first phase of the MADCAT program, our focus was to extend the basic BGM approach to be machine trainable. Through the use of kernel methods and Relevance Vector Machines (RVM), we developed a mathematically rigorous approach to probabilistic graph comparison. We have also started implementing a hyperkernel optimization routine using Brent's method in multiple dimensions to allow automated calculation of hyperkernel parameters.

4.3 Stochastic Segment Models [BBN]

Stochastic segment modeling involves a novel combination of HMMs and 2-D matching approaches such as the bipartite graph matching (BGM). It aims to improve the HMM-based handwritten Arabic text recognition by integrating long-span segment level information with the shorter-span, frame-based information from the HMM. In our current approach, character HMMs that use PACE features are used to force-align training transcriptions to word or line images to automatically generate character boundaries. Next, 2-D images (the stochastic segments) are extracted for each character using these approximate boundaries. Features computed on these 2-D "whole character" images are used to train "segment models". In our current approach, we use support vector machines (SVMs) trained with Gradient and Concavity (GC) features for modeling of stochastic character images.

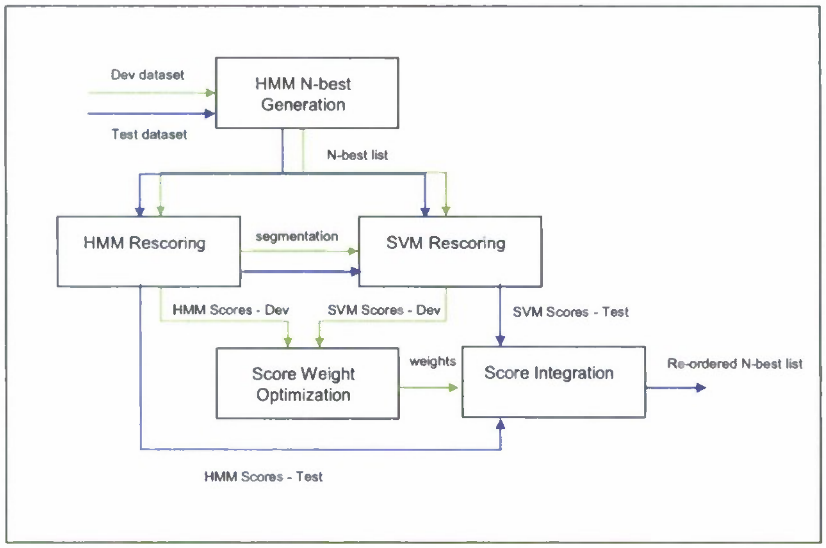


Figure 13: N-best rescoring procedure for using stochastic segment models.

During recognition, first the HMM character models are used to generate segment boundaries for each hypothesis in the n-best list. Each segment is evaluated against the segment models which assigns a probabilistic score. The segment models in our initial implementation are support vector machines (SVMs) trained on stochastic segments in the training images. Finally, the n-best list is rescored by combining the segment model scores with the existing HMM scores, as well as language model scores using weights that are optimized to minimize overall error rate on a development set. Figure 13 is a schematic representation of our current stochastic segment modeling approach. In Table 8, we report on improvements in WER for rescoring n-best lists on the AMA test with the above approach. As shown, using the HMM and the SVM segment scores result in a 2.3% absolute reduction in WER over using only the HMM for rescoring the n-best.

Rescoring Procedure	%WER
HMM only	55.1
HMM + SVM	52.8

Table 8: Stochastic segment based rescoring on AMA test set.

5. Integration with GALE MT [BBN]

The ultimate goal of the MADCAT program is to produce accurate English transcriptions from text in Arabic images. Therefore, we need to integrate the document recognition and understanding capabilities being developed under the MADCAT program with machine translation (MT) and distillation technologies being advanced under the DARPA GALE MT.

In this phase we used BBN's hierarchical MT system (HierDec) being advanced under GALE. Since the MADCAT data consists of a combination of Newswire and Web data, we ran experiments with four different MT systems tuned on either Newswire or Web genre data with and without discriminatively determined corpus weights. The system tuned to web genre with corpus weights out-performed the other three systems on the combined web and newswire test set. We also performed system combination with confusion networks similar to the GALE system. The primary difference between system combination on MADCAT and GALE is that the systems being combined in GALE program come from different sites on the BBN-led AGILE team that use different translation methodologies, whereas for MADCAT system combination, we simply used four different configurations of the BBN HierDec. The system weights were tuned for TERBLEU on a combination of Newswire and Web genre documents from a held-out GALE test set. As shown in Table 9, the combined system outperforms the single-best system across both Newswire and Web genres.

System	Mixed-Case TER	
	Newswire	Web
Single Best System	50.1	56.5
Combined System	49.4	55.7

Table 9: Genre-wise comparison of single best system and combined MT system on error-free text.

Presently, we perform machine translation (MT) on the single-best OCR output. Since the 1-best OCR output has a high error rate and a lattice or n-best is likely to contain the correct answer, we performed an experiment to establish the lower bound for TER by using the best/oracle answer in the OCR n-best as the input to the MT system. As shown in Table 10, for the Devtest Part1a released by LDC, the improvement in translation error rate (TER) for using the oracle n-best hypothesis is modest. Since the oracle hypothesis has a relatively high error rate, we will repeat this experiment with a larger n-best list or a lattice in the next phase.

System	%WER	TER
Error-free text	-	56.4
1-best OCR hypothesis	31.5	65.8
Oracle OCR hypothesis	23.3	63.7

Table 10: Impact of using Oracle n-best hypotheses for translation.

6. Metadata Extraction – Logo Recognition [BAE]

Document logo recognition is a valuable component of an overall document analysis activity. Logo recognition can provide information about the document authors and subject, and can act as a quick pre-screening process for document review. Documents containing certain logos-of-interest can be flagged for review by human analysts. Automated logo recognition can be difficult for several reasons. A number of corrupting factors can degrade logo image quality and thus reduce performance of logo recognition algorithms. Geometric logo misalignments, such as rigid body (translation, rotation, and scale offsets) or more general affine (including skew), can result from improper document copying, mis-calibrated copy machines, and inaccurate logo segmentation. The presence of noise may be an additional problem. Documents that have been duplicated many times, or have been duplicated by degraded copiers may experience salt-and-pepper noise, shading issues, and pixel drop-out. The presence of these contaminating effects can cause difficulties for algorithms based on various technologies. For example, approaches based on geometric features such as corners and shape-based features may be reduced in the presence of noise, which causes the generation of false features. Spot noise, in which pixel blobs can block out small regions of the image, can change the geometric structure of the image thereby reducing matching performance.

In this phase, we performed logo recognition experiments to assess the feasibility of two general approaches; spatial domain correlation approaches, and radon transform domain approaches. Our initial experiments on the UMD logo database and the tobacco logo database indicate that phase correlation-based approaches work better than the radon transform.

We investigated use of the Alpha-Rooted Phase Correlation (ARPC) matching approach in the Fourier log-polar domain. The motivation for this was to perform translation, scale, and rotation-invariant logo recognition. We investigated tolerance of Fourier log-polar domain matching as a function of logo rotation. It was determined that a combination of the interplay between digital sampling and rotation created artifacts that degraded recognition performance. We also determined that error in the cartesian to polar coordinate conversion was a secondary error source which degraded performance. We subsequently built in rotational tolerance by including rotated reference logos associated to each reference logo, and regenerated verification performance results. We also revisited application of the spatial domain ARPC approach to recognition over the UMD logo database. We found that the primary reason for degradation of the spatial domain ARPC approach is the deterioration of the self match score in the presence of logo alignment errors. The addition of the rotational tolerance improved the overall verification performance by increasing self-match scores. We also investigated tolerance of the spatial domain ARPC approach to logo rotation. We defined rotational robustness metrics and determined values of the alpha-rooting parameters that optimized the metrics.

With the reception of additional logo imagery, we revisited the radon transform domain ARPC approach to logo recognition. We discovered that the artifacts due to logo image padding and the rectangular shape of the image bounding box degraded recognition performance. Recognition performance was significantly dependent on the degree of artifacts present in the radon transform domain. We developed an algorithm to estimate appropriate cropping of the radon transform image to reduce the presence of artifacts. We investigated the rotational tolerance of the cropped radon transform and found that appropriate artifact reduction can produce reasonable rotation tolerance. We generated logo verification performance results over the UMD logo database, in the form of Receiver Operating Characteristics (ROC) curves.

Finally, we developed reverse videoing of the pixel values for removing the artifacts caused by Radon transform. In addition, we developed a translation normalization approach and a scale estimation algorithm in the radon transform domain. Together with the rotational invariance of the radon transform and the use of ARPC, we developed a recognition approach invariant to logo translation, rotation, and scale.

7. Evaluation System and Accomplishment of Phase 1 Goals [BBN]

We participated in the MADCAT Phase 1 evaluation held in September 2008. The glyph model used in the evaluation system was trained on a total of 8253 images from 58 different authors. Position-dependent tied mixture (PDTM) HMM models were trained for a total of 176 unique characters. A trigram language model trained on 90 million words of the GALE corpus in combination with a 92K dictionary was used for recognition. Recognition was performed using a two-pass search strategy. The resulting n-best list was then re-ranked using a combination of the acoustic scores, and a language model score which does not model the "white space" token. The weights for re-ranking were tuned on the development set. The top best hypothesis from the re-ranked n-best list was used to adapt the means of the HMM model via MLLR estimation. We trained two text recognition systems – one trained on PACE features, and the other trained on the PACE and gradient and concavity (GC) features, referred to as the GCPACE system.

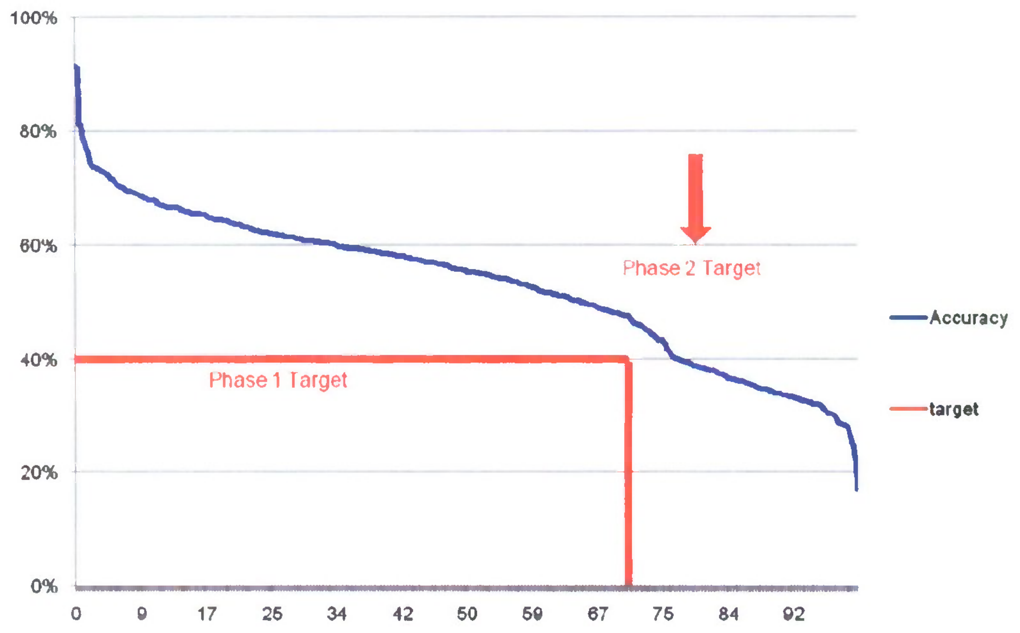


Figure 14: Plot of Accuracy versus Percentage of documents on Phase 1 Evaluation Set.

The MT system used for translation was the BBN's hierarchical MT engines. The best translation output on error-free text was obtained by combining the MT decoding results from four differently tuned systems.

System combination works best when the systems being combined are sufficiently different. Therefore in order to introduce as much variability in the systems being combined as possible, we mix-and-match the two text recognition systems and four MT systems in order to produce four final system outputs for combination. The details are shown in Table 11. The combined system out-performed the single best system giving a relative gain of 4.3% in mixed-case TER.

OCR System	MT System	%WER	Mixed-case TER	Mixed-case BLEU	METEOR
GCPACE	Web, Corpus Weights On	31.5	65.6	18.4	45.5
PACE	Web, Corpus Weights Off	34.4	67.0	17.7	44.0
PACE	Newswire, Corpus Weights On	34.4	67.3	17.1	43.8
GCPACE	Newswire, Corpus Weights Off	31.5	66.5	17.5	45.2
System Combination			62.8	19.5	46.0

Table 11: Phase 1 evaluation system results on text recognition output of test set.

The program metric for MADCAT is human translation error rate (HTER), which involves post-editing the system output by human annotators. The HTER measured on our MADCAT Phase 1 evaluation output is shown in Figure 14. As can be seen from the figure, our team successfully accomplished the Phase 1 targets for MADCAT and we have a modest head-room towards the Phase 2 targets.

8. Analysis and Future Work [BBN]

Periodic analyses of causes for errors are a part of our technical plan for MADCAT. In Phase 1, we developed a comprehensive error analysis methodology for understanding the causes of errors for text recognition. Our preliminary results indicate the following are the four main causes of errors:

1. Poor legibility: significant fraction of words on the page are difficult to read
2. Overlapping words/lines: page is too crowded resulting in words from adjacent lines are touching each other
3. Ruled page-lines: presence of ruled page-lines on the image
4. Skew: the baseline within the text line exhibits varying orientation

For Phase 2 of the effort, we have created a technical plan to address the above causes of errors in our MADCAT system. We also plan on performing a more detailed analysis to refine the technical plan for Phase 2.